LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-JRNL-200451

Stochastic Kinetic Monte Carlo Algorithms for Long-range Hamiltonians

D. R. Mason, R. E. Rudd, A. P. Sutton

September 26, 2003

Submitted for publication in Computer Physics Communications

DISCLAIMER

Stochastic Kinetic Monte Carlo algorithms for long-range Hamiltonians

D. R. Mason

Department of Materials, Oxford University, OX1 3PH, UK

R.E. Rudd

Lawrence Livermore National Laboratory, 7000 East Ave., L-045, Livermore, CA 94550, USA

A.P. Sutton

Department of Materials, Oxford University, OX1 3PH, UK

Helsinki University of Technology, Laboratory of Computational Engineering, PO. Box 9203, FIN 02015 ESPOO, FINLAND.

Abstract

We present a higher order kinetic Monte Carlo methodology suitiable to model the evolution of systems in which the transition rates are non- trivial to calculate or in which Monte Carlo moves are likely to be non- productive flicker events. The second order residence time algorithm first introduced by Athènes et al.[1] is rederived from the n-fold way algorithm of Bortz et al.[2] as a fully stochastic algorithm. The second order algorithm can be dynamically called when necessary to eliminate unproductive flickering between a metastable state and its neighbours. An algorithm combining elements of the first order and second order methods is shown to be more efficient, in terms of the number of rate calculations, than the first order or second order methods alone while remaining statistically identical. This efficiency is of prime importance when dealing with with computationally expensive rate functions such as those arising from long- range Hamiltonians. Our algorithm has been developed for use when considering simulations of vacancy diffusion under the influence of elastic stress fields. We demonstrate the improved efficiency of the method over that of the n-fold way in simulations of vacancy diffusion in alloys. Our algorithm is seen to be an order of magnitude more efficient than the n-fold way in these simulations. We show that when magnesium is added to an Al-2at.%Cu alloy, this has the effect of trapping vacancies. When trapping occurs, we see that our algorithm performs thousands of events for each rate calculation performed.

Preprint submitted to Elsevier Science *26 September 2003*

1 Introduction

There are innumerable instances where a material evolves in time through many competing internal stochastic processes. Each process occurs at a certain average rate, which may either be constant in time, or dependent on how the system has evolved up to that time. Kinetic Monte Carlo methods [3] are standard means of modelling problems such as these, especially when one wishes to model the evolution of the system over periods of time very much longer than those accessible by direct simulation such as molecular dynamics. Using these methods one can simulate the evolution of the internal microstructure of a material over seconds or millenia, depending on the rates of the individual processes. A strength of the kinetic Monte Carlo approach is that the system evolves through defined stochastic processes, in contrast to phase field methods where such processes are not usually specified explicitly. This is particularly significant if one is concerned with accurate modelling of the kinetics of microstructural evolution rather than just the evolution of its morphology.

The methodology presented here is motivated by the wish to model microstructural evolution in alloys through diffusion by an atomic vacancy mechanism. Rautiainen and Sutton [4,5] compared results obtained with kinetic Monte Carlo simulations based on a vacancy diffusion mechanism with those obtained by a phase field approach for phase separation and coarsening in rigid two- dimensional lattices. They found that at low temperatures the microstructures and kinetics of phase separation were quite different in the two simulation methodologies, as was the mechanism of coarsening, even though the atomic interactions were identical in the two cases. These differences were attributed to the trapping of the vacancy at interfaces in the microstructure in the kMC simulation, leading to faceting and a coarsening mechanism in which second phase particles undergo Brownian motion which facilitates agglomeration. A similar conclusion was reached by Weinkamer and Fratzl [6]. Thus the atomic-scale mechanism of mass transport can directly influence both the kinetics and the morphology of the microstructure.

More recently we have modelled the formation of GP zones in Al-Cu alloys in three dimensions by a vacancy diffusion mechanism including long- range elastic interactions between all atoms. These simulations present two principal challenges. First one has to devise a rapid, accurate method of calculating the change in the total elastic energy of the system for trial moves of a vacancy to each of its nearest neighbour sites. Second, one has to overcome the problem of *flickers*, in which a vacancy spends a great deal of time jumping back and forth between neighbouring sites. A common example of a system prone to flickering is an association between a vacancy and one or more impurities forming a complex. In that case exchanges between the vacancy and the impurity comprising the complex may occur far more frequently than movements of the complex as a whole. The problem of flickering is familiar in any kinetic Monte Carlo simulation where the system can become trapped in a metastable equilibrium[7–9]. In this paper we find more than a thousand flickers may occur in Al-2 at.%Cu- 1 at.%Mg alloys between productive moves of a vacancy. To be most useful a solution to the flicker problem must enable an efficient parallel simulation in which several vacancies are allowed to diffuse through the system. The sequence of moves of differ-

ent vacancies must be known in order the for the effects of any one jump to be felt at the appropriate time by all other vacancies.

The purpose of this paper is to describe a stochastic algorithm that overcomes the problems that arise from flickering, the long-range nature of the Hamiltonian and keeping track of the sequence of all events. Our algorithm is also parallelisable. Although our presentation of the method is in terms of modelling vacancy diffusion, it is applicable to other processes where unproductive jumps between adjacent states reduce the efficiency of kinetic Monte Carlo simulations.

There is a glossary at the end of this paper containing definitions of terms which appear in italics.

2 The n-fold Way: a first order residence time Monte Carlo (RTMC) algorithm

Consider a single vacancy in a face- centred cubic metal. It has 12 nearest neighbour sites. Let the site at which the vacancy is located be site I, and the state of the system be i. The state is represented by a point in configuration space corresponding to the particular configuration of all atoms in the system. Let the nearest neighbour sites be labelled J. When an atom at the J site jumps into the vacant site we say that the vacancy has jumped from site I to site J. The system has moved from state i to state j. Let $r_{\mathrm{i}\to\mathrm{j}}$ be the rate at which the vacancy jumps from site I to site J. Equivalently this rate is the rate at which the vacancy jumps from state i to neighbouring state j. This rate is the inverse of the average time taken for an ensemble of systems in state i to move to state j. The probability that the vacancy will jump to site J is given by

$$p_{\mathrm{i}\to\mathrm{j}} = \frac{r_{\mathrm{i}\to\mathrm{j}}}{R_{\mathrm{i}}} \tag{1}$$

where $R_{\mathrm{i}} = \sum_{k=1}^{C} r_{\mathrm{i}\to\mathrm{k}}$ is the sum of the rates of all C processes out of state i. In this case R_{i} is the sum of the rates of the vacancy exchanging places with any of its neighbouring atoms. The probability that the vacancy will jump to any of its neighbouring sites during the period between Δt and $\Delta t + d\Delta t$ is given by

$$p_i\left(\Delta t\right) d\Delta t = R_{\mathrm{i}} \exp\left(-R_{\mathrm{i}}\Delta t\right) d\Delta t \tag{2}$$

In the n-fold way algorithm of Bortz et al.[2] the site to which the vacancy jumps is selected according to the probability prescribed by equation (1). Equation (2) may be inverted to yield $\Delta t = -\ln\left(p_i/R_{\mathrm{i}}\right)/R_{\mathrm{i}}$. To capture the stochastic nature of the process for an *individual* vacancy, as opposed to the ensemble of systems implied in equation (2), time is incremented according to the following expression:

$$\Delta t = -\frac{\ln\zeta}{R_{\mathrm{i}}} \tag{3}$$

where ζ is a uniformly distributed random variable such that $0 < \zeta \leq 1$. Since Δt is the time that elapses before the vacancy moves from site I, Δt is called the *residence time*. Since the rates out of only one state are considered the algorithm is called first order, and the jumps are called first order jumps.

A procedure for incrementing time that has been used elsewhere is to use equation (2) to calculate an average value of Δt, that is $1/R_{\mathrm{i}}$. This quantity is an ensemble average. For an individual vacancy incrementing the time according to equation (3) reflects more accurately the stochastic nature of its motion. But the more substantive criticism of using $\Delta t = 1/R_{\mathrm{i}}$ arises when one wishes to model two or more vacancies in an inhomogeneous system. If the movement of the vacancies is correlated owing to a long range elastic interaction between them, then it is necessary to define a single synchronous time for all vacancies. This becomes a key point when parallelisation strategies based on domain decomposition are considered. [10–13] If vacancies are allowed to move independently on different processors in a race condition then stochastic time increments must be used to ensure causality may be strictly imposed.

The n-fold way treats two or more vacancies in a manner that allows time to be incremented in a truly stochastic manner by the same amount for all vacancies. One evaluates R_{i} as the sum of the rates of all vacancy- atom exchanges in equation (1) and equation(3). The problem with the n-fold way algorithm is that at a metastable equilibrium by far the most probable vacancy jump may be to reverse the jump that had just been made. In that case the vacancy flickers back and forth to adjacent sites many times before moving on, and the simulation becomes very inefficient.

3 The second- order residence time algorithm of Athènes et al.[1]

In this section we rederive the second order algorithm of [1] introducing some essential new terms and notation for use in subsequent sections. In particular, this will set the scene for a new stochastic implementation of the second order residence time algorithm in the next section.

To remove unproductive jumps we must consider a sequence of jumps from the current state. If we look at a sequence of m jumps from the current state the algorithm is called m'th order. The second order residence time algorithm of Athènes et al.[1] was developed to overcome the problem of the vacancy flickering between adjacent sites. They sought the probability that the system moves from its current state to a particular new state but then does not immediately jump back. This may take place by the system jumping from state j to state k and then from state k to state l, where $\mathrm{l} \neq \mathrm{j}$. Such a jump from state j to state k which then escapes we shall term a *direct* move. However, if state j is a metastable state, then before moving away in this manner, the system may undertake a series of pairs of jumps to and fro between state j and immediately neighbouring short-lived states k. A pair of consecutive jumps j→k and k→j defines a *flicker* between states j and k. The move from state j to state

k which includes one or more flickers before finally escaping with a jump from state k to state l$\neq$ j we term an *indirect* move.

The probability of a direct or indirect move from any starting state j will differ from the probability of a jump in the n-fold way owing to flickers. At the end of each move we know that the next jump may not be a reversal, so our probability will also be explicitly dependent on the now excluded probability of jumping to the previously visited state.

We must know more rates than were required in the n- fold way algorithm to exclude flickers, as transition probabilities for both the forward and backward jumps must be known. We must also keep track of which state was previously visited.

Let the system be moved from state i to state j. In our system this means that the vacancy has been moved from site I to site J. We define *event i* to be the jump that takes the system from whatever state it currently occupies to state i. A *reversal event* takes the system back to the previous state, if one is defined. The probability that the first event after the system arrives at state j is to state k, given that we have excluded the reversal to state i, is given by

$$p_{\mathrm{j}\to\mathrm{k}(i)} = \frac{p_{\mathrm{j}\to\mathrm{k}}}{1-p_{\mathrm{j}\to\mathrm{i}}}\left(1-\delta_{\mathrm{ki}}\right) \tag{4}$$

where δ_{ki} is the Kronecker delta. The denominator in this posterior probability is required for renormalisation. The transition probabilities $p_{\mathrm{j}\to\mathrm{k}}$ are given by the n-fold way in equation (1).

Notice that equation (4) requires us to know which was the previously visited state i. If we have not made any moves previously, and the model has been initialised by some arbitrary means, then this previous state i will not be defined. We could take care of this by simply using the n-fold way probability of equation (1) in place of equation (4) for this first call. A notational and coding trick, which is also exploited in section 6, is to define a null state 0, which has the property that $r_{\mathrm{j}\to 0} = 0$ for all j. We can then set the 'previous' state to be this null state 0, and since the next state visited, state k, will never be state 0, equation (4) is seen to recover the correct transition probabilities, i.e. $p_{\mathrm{j}\to\mathrm{k}(i)} = p_{\mathrm{j}\to\mathrm{k}}$.

A useful quantity to derive is the probability that the first two events after arriving in state j comprise a flicker j $\to$ k $\to$ j, where state k $\neq$ state i. This quantity we will denote $\gamma_{\mathrm{j}(i)}$. We can write down $\gamma_{\mathrm{j}(i)}$ in terms of the transition probabilities in equations (1) and (4).

$$\gamma_{\mathrm{j}(i)} \equiv \sum_k p_{\mathrm{j}\to\mathrm{k}(i)}p_{\mathrm{k}\to\mathrm{j}} = \sum_k \frac{p_{\mathrm{j}\to\mathrm{k}}p_{\mathrm{k}\to\mathrm{j}}}{1-p_{\mathrm{j}\to\mathrm{i}}}\left(1-\delta_{\mathrm{ki}}\right) = \frac{\gamma_{\mathrm{j}(0)}-\gamma_{\mathrm{j}\leftrightarrow\mathrm{i}}}{1-p_{\mathrm{j}\to\mathrm{i}}} \tag{5}$$

Note that $\gamma_{\mathrm{j}(0)} = \sum_k p_{\mathrm{j}\to\mathrm{k}}p_{\mathrm{k}\to\mathrm{j}} = \sum_k \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$, where $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} = p_{\mathrm{j}\to\mathrm{k}}p_{\mathrm{k}\to\mathrm{j}}$. Note that $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} = \gamma_{\mathrm{k}\leftrightarrow\mathrm{j}}$, and that to find $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ it will be necessary to determine the rates of all events accessible from both states j and k.

The probability $p\left(f\right)$ that $f \geq 1$ flickers occur before the system moves on is $\gamma_{\mathrm{j}(i)}\gamma_{\mathrm{j}(0)}^{(f-1)}\left(1-\gamma_{\mathrm{j}(0)}\right)$.

The expected number of flickers is given by

$$\langle f \rangle \equiv \sum_{f'=0}^{\infty} f' p\left(f'\right) = \frac{\gamma_{\mathrm{j}(i)}}{\left(1-\gamma_{\mathrm{j}(0)}\right)} \tag{6}$$

If the first two events after arriving in state j do comprise a flicker, then the move from state j to state k will, by our definition above, be an indirect move. If such a pair does not constitute the first two jumps from state j, then a direct move must have occurred, and so the probability of a direct move is $1-\gamma_{\mathrm{j}(i)}$:

$$p \text{ direct} = \sum_{k} \left(p_{\mathrm{j}\to\mathrm{k}(i)} - p_{\mathrm{j}\to\mathrm{k}(i)} p_{\mathrm{k}\to\mathrm{j}}\right) = \sum_{k} \frac{\left(p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}\right)\left(1-\delta_{\mathrm{ki}}\right)}{1-p_{\mathrm{j}\to\mathrm{i}}} = 1-\gamma_{\mathrm{j}(i)} \tag{7}$$

Since the system may evolve only through direct or indirect moves, the probability of an indirect move occurring is therefore equal to $\gamma_{\mathrm{j}(i)}$:

$$p \text{ indirect} = \sum_{k} \frac{\gamma_{\mathrm{j}(i)}}{1-\gamma_{\mathrm{j}(0)}} \left(p_{\mathrm{j}\to\mathrm{k}} - p_{\mathrm{j}\to\mathrm{k}(i)} p_{\mathrm{k}\to\mathrm{j}}\right) = \gamma_{\mathrm{j}(i)} \tag{8}$$

Thus we obtain the following relation for the probability of a direct or indirect move from state j to state k, which excludes the first jump immediately returning the system to state i:

$$P\left(k\right) = \underbrace{\frac{\left(1-\delta_{\mathrm{ki}}\right)}{1-p_{\mathrm{j}\to\mathrm{i}}}\left(p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}\right)}_{direct} + \underbrace{\frac{\gamma_{\mathrm{j}(i)}}{1-\gamma_{\mathrm{j}(0)}}\left(p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}\right)}_{indirect} \tag{9}$$

4 A stochastic version of the second order residence time algorithm

Once it is established whether the move of the vacancy from site J to site K is direct or indirect, that knowledge alters the probability, from that given in equation (9), of selecting the site which will be occupied next by the vacancy. Nevertheless, it still follows from equation (9) that the probability of an indirect move from state j involving $f \geq 1$ flickers is given by

$$p\left(f\right) = \gamma_{\mathrm{j}(i)} \gamma_{\mathrm{j}(0)}^{f-1} \sum_{k} \left(p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}\right) = \gamma_{\mathrm{j}(i)} \gamma_{\mathrm{j}(0)}^{f-1} \left(1-\gamma_{\mathrm{j}(0)}\right) \tag{10}$$

where $\gamma_{\mathrm{j}(i)}$ is given by equation (5). As we have already noted in equation (7), the probability of a direct move from state j is given by

$$p\left(0\right) = 1-\gamma_{\mathrm{j}(i)} \tag{11}$$

These probabilities satisfy $\sum_{f=0}^{\infty} p(f) = 1$. If we define $\pi(f)$ by

$$\pi(f) = \sum_{f'=0}^{f} p(f') \tag{12}$$

then we find that

$$\pi(f) = 1 - \gamma_{\mathrm{j}(i)} \gamma_{\mathrm{j}(0)}^{f} \quad \text{where } f \geq 0. \tag{13}$$

Equation (13) may be inverted to yield an expression for the number of flickers in terms of a uniform random variable $0 < \zeta \leq 1$ representing $1 - \pi(f)$:

$$\begin{aligned} f &= \mathrm{int}\left(\frac{\ln\left[\zeta/\gamma_{\mathrm{j}(i)}\right]}{\ln \gamma_{\mathrm{j}(0)}}\right) + 1 \quad \text{if } \zeta \leq \gamma_{\mathrm{j}(i)} \\ &= 0 \qquad\qquad\qquad\qquad\quad \text{if } \zeta > \gamma_{\mathrm{j}(i)} \end{aligned} \tag{14}$$

where the function $\mathrm{int}(x)$ returns the largest integer smaller than or equal to x. Once it is determined whether the next move is direct or indirect the probability, $P(m)$ in equation (9), that m is the next state becomes:

$$P(m) = \begin{cases} \dfrac{p_{\mathrm{j}\to\mathrm{m}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{m}}}{1 - p_{\mathrm{j}\to\mathrm{i}} - \gamma_{\mathrm{j}(0)} + \gamma_{\mathrm{j}\leftrightarrow\mathrm{i}}} (1 - \delta_{\mathrm{mi}}) & \text{for } f = 0 \text{ i.e. a direct move} \\ \dfrac{p_{\mathrm{j}\to\mathrm{m}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{m}}}{1 - \gamma_{\mathrm{j}(0)}} & \text{for } f > 0 \text{ i.e. an indirect move} \end{cases} \tag{15}$$

If the move is indirect the number of flickers to each state k neighbouring j has to be found, such that the total number of flickers is given by the first line of equation (14). The state k_1 of the first flicker will be chosen from the distribution

$$p(\mathrm{k}_1) = \frac{\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}_1}}{\gamma_{\mathrm{j}(0)} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{i}}} (1 - \delta_{\mathrm{k}_1\mathrm{i}}) \tag{16}$$

This leaves $f - 1$ flickers to be distributed among all the nearest neighours of state j. Select any state $\mathrm{k} = \mathrm{k}_2$ neighbouring state j. The probability, $d(q)$, of $q \leq f - 1$ flickers to state k_2 is determined by the binomial distribution:

$$d(q) = \frac{\phi!}{q!\,(\phi - q)!} \left(p(\mathrm{k}_2)\right)^q \left(1 - \left(p(\mathrm{k}_2)\right)\right)^{(\phi - q)} \tag{17}$$

where $\phi = f - 1$ and $p(\mathrm{k}_2) = \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}_2}/\gamma_{\mathrm{j}(0)}$.

Selection of a number $q = q_{\mathrm{k}_2}$ from this binomial distribution may be done using standard rejection sampling techniques. However if ϕ is small, it is easier to directly sample, and if

$\phi p\,(\mathrm{k})$ is small the distribution is quite accurately Poisson. An algorithm for selection from the binomial distribution under these circumstances is given in [14]. This leaves $f-1-q_{\mathrm{k}_2}$ flickers to be distributed among the remaining states. Select another state $\mathrm{k}=\mathrm{k}_3\neq\mathrm{k}_2$ neighbouring state j for the next flicker. Setting $\phi=f-1-q_{\mathrm{k}_2}$, the probability of $q\leq\phi$ flickers to state $\mathrm{k}=\mathrm{k}_3$ is again determined by the binomial distribution of equation (17), with $p\,(\mathrm{k}_2)$ replaced by $p\,(\mathrm{k}_3)=\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}_3}/\left(\gamma_{\mathrm{j}(\mathit{0})}-\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}_2}\right)$. A number $q=q_{\mathrm{k}_3}$ of flickers to state $\mathrm{k}=\mathrm{k}_3$ is selected according to this distribution. The algorithm is repeated until there are no more flickers to be allocated among the remaining nearest neighbours of state j, or there is just one neighbouring state left which is then allocated all the remaining flickers. Finally the value of q_{k_1} is incremented by one to account for the first flicker. In this way the algorithm provides the number of flickers to each state neighbouring state j, such that the total number of flickers is f. The maximum number of times a binomial distribution is sampled is $C-1$. For our example of a single vacancy diffusing this number will be $z-1$, where z is the coordination number, regardless of the actual number of flickers which may occur. Having determined the number of flickers $q_{\mathrm{k}'}$ to each state k' neighbouring state j before the vacancy makes a direct move, we may evaluate the time taken for the vacancy to complete this move. Given that the residence time of the system in state j is determined by the exponential distribution of equation (2), the probability that the vacancy will jump n times from site J during the time interval between Δt and $\Delta t+d\Delta t$ is given by the gamma distribution [14]:

$$p\,(\Delta t)\,d\Delta t=\frac{R_{\mathrm{j}}^{n}\,(\Delta t)^{(n-1)}}{(n-1)!}\exp\left(-R_{\mathrm{j}}\Delta t\right)d\Delta t \tag{18}$$

The gamma distribution can be sampled, again using rejection sampling, to give a stochastic time, $\Delta t_n^{(\mathrm{j})}$, for the system to make n jumps out of state j. It follows that the total time taken for the vacancy to complete the move from site J is given by:

$$\Delta t=\Delta t_{f+1}^{(\mathrm{j})}+\sum_{k'}\Delta t_{q_{\mathrm{k}'}}^{(\mathrm{k}')} \tag{19}$$

where $q_{\mathrm{k}'}$ is the number of flickers to state k' neighbouring state j. The first term in equation (19) describes the residence time at state j during which f flickers occur. If $f=0$ then the first term is the first order residence time at state j. The second term in equation (19) describes the residence time at each state neighbouring state j.

Let us summarise the stochastic second order residence time algorithm:

(1) First determine the number of flickers from the site currently occupied by the vacancy using equation (14), and hence whether the next move is direct or indirect.
(2) Determine the site neighbouring the current site to which the vacancy will move using equation (15).
(3) If the move is indirect determine the number of flickers to each neighbouring site using equations (16) and (17).

(4) Determine the residence time at the current site and each neighbouring site using equation (18).
(5) Calculate the total time for the vacancy to complete its move using equation (19).

5 Scaling problems with the second order algorithm

The second order algorithm introduced in the previous section allows us to determine a stochastic residence time for a direct or an indirect move. However the algorithm in the form described above is often less efficient than the n-fold way.

Firstly, not all moves are necessarily independent. In our example of vacancy diffusion moving one vacancy affects the rates at which other vacancies move because of their long-range elastic interactions. The number, C, of neighbouring states of the system is vz, where v is the number of vacancies and z is the lattice coordination number. C rates must be known before the next event to occur can be determined. To evaluate each of the $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$, it is necessary to find (a) the probability $p_{\mathrm{j}\rightarrow\mathrm{k}}$, which involves C rates out of state j, and (b) all the C rates of processes out of each state k. Therefore to determine the next event using the second order algorithm as written above requires finding $C \times (C+1)$ rates. So now the computational time required to determine the next event depends on the square of the number of events.

A second problem will also arise when the number of events increases. The probability of the next pair of events comprising a flicker $j \rightarrow k \rightarrow j$ is inversely proportional to the sum of rates out of state k. As the number of events increases, this sum gets larger, and so the probability of a flicker at generic sites is reduced, even though the rate of the reversal event remains the same.

6 Combining first and second order algorithms

We have developed an algorithm which remains efficient by combining both first- and second-order algorithms. This new algorithm uses a record of previously visited states to eliminate flickers in a more targeted manner. In the following we continue to assume that the current state is j, and the next state is k or k′.

To move from a first order to a second order algorithm, we included extra information in the form of the probability that a pair of events comprise a flicker, $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$, and the previously visited state i. Consider removing this information from the second order algorithm by setting all the $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ to zero, and state i to the null state 0, as if we were restarting the simulation. Then the sum $\gamma_{\mathrm{j}(0)} = \sum_k \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} = 0$, $p_{\mathrm{j}\rightarrow\mathrm{i}} = r_{\mathrm{j}\rightarrow\mathrm{i}} = r_{\mathrm{j}\rightarrow 0} = 0$ and the next visited state $\mathrm{k} \neq \mathrm{i}$. The number of flickers (given by equation (14)) will always be zero as $\gamma_{\mathrm{j}(i)}$=0, and so the next move will always be direct. The transition probability from equation (15) now reads $P(m) = p_{\mathrm{j}\rightarrow\mathrm{m}}$, exactly that given by the n-fold way. Finally the residence time will be given

by equation (18) for a single escape- again returning the n-fold way result. By setting all $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ to zero we have effectively recovered the n-fold way, but in doing this we lost the ability to remove flickering. By setting $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ to zero we are not asserting that flickering between states j and k does not occur because if the system moves from state j to k it may then move straight back to state j, which amounts to a flicker.

Now consider combining the two algorithms by setting just some of the $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ to zero. We shall see that this leads to a well-defined intermediate algorithm, allowing much greater flexibility. We still need $p_{\mathrm{j}\to\mathrm{k}}$ for all events but we no longer require all $p_{\mathrm{k}\to\mathrm{k}'}$.

With the current state of the system being state j, we find the flicker probability $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ of one group of events, but we set $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$ to zero for events in a second group. If in moving to state j we considered flickering between states i and j, then we explicitly disallow the immediate reversal j $\to$ i and the rates back and forth between state j and the previous state i are known. State i cannot be one of the states k′ for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'} = 0$. Conversely if in moving to state j we did consider flickering, then we can not disallow the immediate reversal j $\to$ i so there is no excluded reversal event. But we must create one in order to apply equations (14 - 19). This can be done by setting the excluded reversal event to $\mathit{0}$, or equivalently by setting the previously visited state i to the null state 0 as with the initialisation step. Substituting these into equation (9) gives four possibilities

$$\begin{array}{llll}
\text{(a) event } i \text{ defined } \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} \text{ known} & P(k) &= \left[\left(\frac{1-\delta_{\mathrm{ki}}}{1-p_{\mathrm{j}\to\mathrm{i}}}\right) + \frac{\gamma_{\mathrm{j}(i)}}{1-\gamma_{\mathrm{j}(0)}}\right](p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}) \\
\text{(b) event } i \text{ defined } \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'} = 0 & P(k') &= \left[\left(\frac{1}{1-p_{\mathrm{j}\to\mathrm{i}}}\right) + \frac{\gamma_{\mathrm{j}(i)}}{1-\gamma_{\mathrm{j}(0)}}\right] p_{\mathrm{j}\to\mathrm{k}'} \\
\text{(c) i} = 0 \quad \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} \text{ known} & P(k) &= \left[\frac{1}{1-\gamma_{\mathrm{j}(0)}}\right](p_{\mathrm{j}\to\mathrm{k}} - \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}) \\
\text{(d) i} = 0 \quad \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'} = 0 & P(k') &= \left[\frac{1}{1-\gamma_{\mathrm{j}(0)}}\right] p_{\mathrm{j}\to\mathrm{k}}
\end{array} \tag{20}$$

Note that we do not need the term $(1-\delta_{\mathrm{k}'\mathrm{i}})$ in (b). As we know that we have come from state i, we will always have $p_{\mathrm{i}\to\mathrm{j}}$, and so we always have $\gamma_{\mathrm{j}\leftrightarrow\mathrm{i}}$. In (c) and (d) we have used the fact that when the previous state i is the null state, then $\gamma_{\mathrm{j}(i)}$=$\gamma_{\mathrm{j}(0)}$ and so $\left[\left(\frac{1-\delta_{\mathrm{ki}}}{1-p_{\mathrm{j}\to\mathrm{i}}}\right) + \frac{\gamma_{\mathrm{j}(i)}}{1-\gamma_{\mathrm{j}(0)}}\right] = \left[\frac{1}{1-\gamma_{\mathrm{j}(0)}}\right]$.

If a move is selected which takes the system from state j to a state k for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ is known, then we know that the *next* event will not be a reversal k$\to$j. From state k, the next move will be decided using (a) and (b). If however we select a state k′ for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$ has been set to zero, we will not yet have excluded the reversal. From state k′, the previous state will be j, but since the direct move k′ $\to$ j is not excluded we may not use (a) and (b). The next move will be decided using (c) and (d), as if we were restarting the simulation from state j.

We can use the same procedure as section 4 to determine the next move. The move may still be direct or indirect, with the number of flickers still given by equation (14), where now $\gamma_{\mathrm{j}(0)} = \sum_k \gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ is over the remaining flickering states. The next state visited will still be decided with equation (15), and the flickering states and time taken with equations (16 -

19). All that we have done is to generalise $\gamma_{\mathrm{j}(0)}$ to be the probability of a flicker occurring to one of the states k that have been considered, rather than a probability of flickering to any state. Similarly $\gamma_{\mathrm{j}(i)}$ is still the probability of an indirect jump from state j, albeit with a reduced set of states to which flickers may occur.

If we choose to move to one of the states k′ for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$ was set to zero, we might still end up moving from state j→k′→j, effectively producing a flicker. In doing this we would have constructed the transition probabilities required to evaluate $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$. But if a second flicker occurs between these states we may use $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$. It follows that our combined first and second order algorithm is guaranteed not to require more calculations of transition rates than the n-fold way algorithm, and it will require considerably fewer when flickering events become significant.

6.1 Chain RTMC

As the system moves through configurational phase space, some states become accessible while others become inaccessible. At any time there are up to C events which may occur, and each is considered to be independent. However, when one event occurs we see that although the list of accessible states changes, the rate at which some processes may occur will be adjusted by such a small amount that they can for practical purposes be treated as unchanged. In the case of vacancy diffusion in the presence of elastic fields, we might assume that the movement of vacancies separated by more than some distance is uncorrelated, in the sense that a jump of one does not affect the rate of jumping of the other. The distance over which such correlations exist is dependent on temperature, because it depends on the change in the elastic interaction energy as compared with k_BT.

The influence set $\mathcal{K}$ of event k contains all the physical processes whose rates are affected by moving to state k. To find the probability of the next event to occur after moving to state k, it is necessary only to recalculate the rates of events representing the processes in set $\mathcal{K}$.

For a single diffusing vacancy in an inhomogeneous medium, there is only a single influence set comprising all $C = z$ processes, as each event influences all the others. For a pair of diffusing vacancies there are two possibilities. If the pair are close together there will again be only a single influence set, but this time comprising $C = 2z$ processes, as moving one vacancy will influence the direction of movement of the other. If the pair become more widely separated, then we may choose instead to define two influence sets: one for each vacancy containing z members, as a movement of one vacancy will affect the rates of other events involving movement of the same vacancy, but not affect the rates of moving the other.

Exploiting the combination of the first and second order algorithms in section 6, and the possible independence of events leads to our chain RTMC algorithm. This is a further generalisation of the second order residence time algorithm which can be employed to reduce the number of rate calculations required for each event performed for any Hamiltonian to no more than that required by the combined first and second order algorithm.

In the preceding section we had some states k for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ was known, and others k′ for which $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$ was not. In both cases we must have a transition rate $r_{\mathrm{j}\to\mathrm{k}}$ between the current state j and neighbouring states (k or k′). The rate $r_{\mathrm{j}\to\mathrm{k}}$ is required for the first order algorithm and it constitutes the *first order information* of event k. If we have also found the rates from state k, i.e. $r_{\mathrm{k}\to\mathrm{l}}$ for all states l accessible from state k then we have first order information of events from state k, but viewed from state j they constitute *second order information* for the event k.

To apply our chain RTMC algorithm we keep track of which events have reliable rate information associated with them. Starting from state j, we use the selection procedure of section 4 with flicker probabilities $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}'}$ set to zero as in section 6 for all events k' for which only first order information is held. If we have visited state k previously, then we have second order information for event k as we know the rates of all events accessible from state k and so we have $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$. Only the rates $r_{\mathrm{k}\to\mathrm{l}}$ of events l which are members of the influence set of event k need to be stored.

It is also possible to determine the next event to occur without any further calculation if an event k with second order information is selected, as the transition rates are still known for each event. The rate of an event m which is not a member of the influence set of event k is unaffected and so $r_{\mathrm{j}\to\mathrm{m}} = r_{\mathrm{k}\to\mathrm{m}}$. Any second order information for event m is similarly unaffected. So long as one of these is selected, or if the reverse event occurs, a chain of events can be extended indefinitely without further rate calculation.

If a history of recently visited states is kept, then it is the events to states which have previously been visited for which second order information is held. The system may flicker between any of the local states k near state j. When an event k' to a previously unvisited state k′ is selected, the rates to accessible states neighbouring state k′ are not known. When such an event is selected, only first order information is held and so the chain must be stopped and recalculation of rates performed.

Extra book-keeping is required to keep track of the level of information associated with each event. There are five possible levels for an event k, starting from a state j:

Level 0 The rate $r_{\mathrm{j}\to\mathrm{k}}$ has not yet been calculated. It is not possible to determine which is the next event to occur if one or more events have an information level of 0.

Level 1a The rate $r_{\mathrm{j}\to\mathrm{k}}$ is known, but as the subsequent rates $r_{\mathrm{k}\to\mathrm{l}}$ are not, the flicker probability $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ can not be found. We must set $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ to zero.

Level 1b The rate $r_{\mathrm{j}\to\mathrm{k}}$ is known, and stored in memory are rates $r_{\mathrm{k}\to\mathrm{l}}$ and a value for $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$. However these values have become invalid since an event m occurred which had event k in its influence set. If and only if the reversal event to state m is later performed can the second- order information and flicker probability for this state be reinstated.

Level 2a $r_{\mathrm{j}\to\mathrm{k}}$ and the subsequent rates $r_{\mathrm{k}\to\mathrm{l}}$ for all physical processes in event k's influence set have been calculated and stored. $\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}}$ is therefore known. If event k is selected the rates of all events l representing physical processes in k's influence set can be unpacked, and another selection can be made. However, in doing this the level of information held for events l is set to 1b.

	new information level				
event k selected	influenced event l was level				reversal event
had information	1a	1b	2a	2b	
1a	0	0	0	0	0
1b	0	0	0	0	0
2a	1a	1a	1b	X	2b
2b	1a	2a	X	X	2a

Table 1
Information update strategy. All events l whose rates are affected by an event k happening have their information level updated to the values in this table. An X indicates that this situation should never occur.

Level 2b This event corresponds to a reversal, undoing a previous event. The rate $r_{\mathrm{j}\to\mathrm{k}}$ and all the subsequent rates $r_{\mathrm{k}\to\mathrm{l}}$ for all events representing physical processes in common influence sets to k have been found. If this event is selected, not only are the rates $r_{\mathrm{j}\to\mathrm{k}}$ held in memory, but also the rates $r_{\mathrm{k}\to\mathrm{l}}$ in k's influence set with level $1b$ information are reinstated.

The strategy for updating the information level of neighbouring events is shown in table 1. Events which have an information level of 2a or 2b have a flicker probability. If such an event is selected then a reversal is defined and rates are updated using those stored in memory and another selection can be made. If an event with an information level 1a or 1b is selected, the chain must be halted, as at least one event will now have an information level 0. The system can be evolved along the chain without having to compute any further rates until the end of the chain is reached. It is necessary to update the system physically only to the state at the end of the chain, but the residence times of all intermediate states are available for the kinetics of the process and any statistics one wishes to generate about the path along which the system evolves. Once the end of the chain is reached the required new rate calculations can be carried out in a trivially parallelisable manner. Note that since the last event in the chain must always have had no flicker probability, because it was an event with information level 1a or 1b, there will never be any excluded reversal event defined at the end of the chain. The calculation of rates between successive chains will reinitialize the algorithm, as described in section 3.

7 Approximation Schemes and Error Correction

If the transition rates require significant calculation, as might be the case if a long- ranged Hamiltonian were being employed, it is sometimes possible to use approximate rates to determine the path along which the system evolves. In this section we describe how errors introduced by approximations can be partially corrected within the chain RTMC algorithm, and how more precise calculations can be used to keep track of errors.

The rate at which the system moves between states j and k is a function only of the difference between the free energy at j, g_{j}, and the free energy of the saddle point, $g_{\mathrm{j}\leftrightarrow\mathrm{k}}$ between states j and k:

$$r_{\mathrm{j}\to\mathrm{k}} = f\left(g_{\mathrm{j}\leftrightarrow\mathrm{k}} - g_{\mathrm{j}}\right) \tag{21}$$

Suppose the free energy of the system in any state may be evaluated to any required precision, but that a good approximate scheme has been determined from the physics of the problem. We will use approximate free energies to determine the path taken by the system, but then once chosen we will perform precise calculations to check the validity of the path. Specifically, after having moved from state i to state j, we ensure precise values are held for the free energy of state j, g_{j}, and the saddle point $g_{\mathrm{i}\leftrightarrow\mathrm{j}}$. Values generated by our physically acceptable approximation will be labelled with a caret ($\hat{}$), for instance the approximate free energy of the saddle point $g_{\mathrm{j}\leftrightarrow\mathrm{k}}$ will be written $\hat{g}_{\mathrm{j}\leftrightarrow\mathrm{k}}$.

To make a move we need to find the approximate free energy of saddle points between state j and neighbouring states k. The error of the approximation consists of a systematic component s_{j}, which might depend on the path taken to arrive at state j and may be large, and a small additional component $\epsilon_{\mathrm{j}\leftrightarrow\mathrm{k}}$.

7.1 First Order Error Correction

We should always have available precise free energies for g_{i}, g_{j} and the saddle point $g_{\mathrm{i}\leftrightarrow\mathrm{j}}$, so we can write down an estimate for the systematic component of the error for events out of state j:

$$s_j \approx g_{\mathrm{j}\leftrightarrow\mathrm{i}} - \hat{g}_{\mathrm{j}\leftrightarrow\mathrm{i}} \tag{22}$$

where we note that $g_{\mathrm{j}\leftrightarrow\mathrm{i}} = g_{\mathrm{i}\leftrightarrow\mathrm{j}}$.

Using this estimate for the systematic error gives a second, better, approximation for the free energy of the saddle point energy:

$$\tilde{g}_{\mathrm{j}\leftrightarrow\mathrm{k}} \equiv \hat{g}_{\mathrm{j}\leftrightarrow\mathrm{k}} + s_j = \hat{g}_{\mathrm{j}\leftrightarrow\mathrm{k}} + g_{\mathrm{j}\leftrightarrow\mathrm{i}} - \hat{g}_{\mathrm{j}\leftrightarrow\mathrm{i}} \tag{23}$$

This corrected approximation has a much smaller error given by the difference between the errors: $\tilde{g}_{\mathrm{j}\leftrightarrow\mathrm{k}} = g_{\mathrm{j}\leftrightarrow\mathrm{k}} + (\epsilon_{\mathrm{j}\leftrightarrow\mathrm{k}} - \epsilon_{\mathrm{j}\leftrightarrow\mathrm{i}})$

At state j, we have to find the free energies of all accessible saddle points $g_{\mathrm{j}\leftrightarrow\mathrm{k}}$. After searching the lookup table for previously calculated free energies, and having approximated the remainder, a set of free energies is available, each of which has one of three levels of reliability.

reliability level 1 the calculation was approximate only, and the free energy is $\hat{g}_{\mathrm{j}\leftrightarrow\mathrm{k}}$.

reliability level 2a the calculation was approximate only, but has been corrected to remove the systematic error $\tilde{g}_{\mathrm{j}\leftrightarrow\mathrm{k}}$.

reliability level 2b the calculation was exact, and the free energy is $g_{\mathrm{j}\leftrightarrow\mathrm{k}}$.

It is acceptable to compare free energies of reliability levels 2a and 2b if the random component of the error in an approximate calculation is small compared to the thermal energy. It is unacceptable to compare directly a level 1 approximation with a level 2a or 2b because the systematic error in the level 1 calculation has not been removed. However, as shown above, if reliability level 1 approximations have been found, it is always possible to find the systematic component of the error in at most one further approximate calculation.

The random component of the error made in the approximation is also always available. After we have selected a particular event to occur (say k), we have an approximate saddle point energy $\hat{g}_{\mathrm{k}\leftrightarrow\mathrm{j}}$. We also find the same saddle point energy precisely from state k, $g_{\mathrm{k}\leftrightarrow\mathrm{j}}$. The difference between the two gives an estimate of the random component of the error, $\epsilon \approx g_{\mathrm{k}\leftrightarrow\mathrm{j}} - \hat{g}_{\mathrm{k}\leftrightarrow\mathrm{j}}$. If this error exceeds some prescribed tolerance dependent on the temperature, the move can be discarded, and more accurate calculation substituted for the approximations.

7.2 Chain Algorithm Error Correction

In constructing a chain of events, flickers were considered between state j and any previously visited accessible metastable states k. For each of these previously visited states k it follows from section 7.1 that the free energy of states accessible from these are known reliably. We can therefore be confident about the predicted number of flickers for each move in the chain, provided an exact calculation is performed after each move. As the chain algorithm will continue generating moves until the system moves to a state not previously visited, this exact calculation will be required only at the end of the chain.

At the end of the chain an estimate for the energy of the last saddle point traversed will be available. If we have assumed that some of the physical processes which occurred belonged to different influence sets, it is now possible to test that assumption. After an exact calculation for the energy of the last saddle point is performed, we can look at the error in the energy of the last saddle point. If this error is too great compared to $k_B T$, we must reject this last move. This process of unpicking moves can be continued until the chain has only a single move if necessary.

This can be wasteful of computational time, but as each exact calculation is stored in the history of visited states, it has a chance of being reused. In practice it will be better to prescribe independence of events only when such unpicking is likely.

	atomic %		
alloy	Al	Cu	Mg
1	100	0	0
2	98.0	2.0	0
3	97.8	2.0	0.2
4	97.5	2.0	0.5
5	97.0	2.0	1.0

Table 2
Alloy compositions used in the simulations

8 Results

We have developed the chain RTMC algorithm to simulate diffusional phase transformations effected by a vacancy mechanism, where the entire system is kept atomistically relaxed at all times. To demonstrate the efficiency of the chain RTMC method we consider the case of vacancy diffusion in Al-Cu-Mg alloys on a *rigid* lattice because the absence of atomic relaxation enables us to perform a sufficient number of simulations to generate statistically meaningful results. We expect similar computational gains when the chain RTMC algorithm is implemented with full atomic relaxation.

8.1 Model of vacancy diffusion in Al-Cu-Mg alloys

We have constructed a set of relatively short-ranged Finnis-Sinclair potentials [15] for this study. They have been parameterised to reproduce elastic constants, the cohesive energy and lattice parameters of the pure metals. Appendix A describes in detail the potentials we have constructed and used in these simulations.

The alloy systems simulated comprised 10976 lattice sites (14x14x14 fcc unit cells), with 2 vacancies and a simulated temperature of 500K. The 256 most recently visited states were stored so that the chain RTMC algorithm could determine flickering events. The alloy compositions used are listed in table 2.

An event in this system is the exchange of one vacancy with an atom in one of the twelve nearest neighbouring positions. The rate appropriate for the movement of a point defect is given by Vineyard's equations[16]. While the terms in this expression could be calculated exactly, in practice it is difficult to establish converged saddle point energies 'on the fly'. Instead we have used Flynn's approximation [17] derived from dynamical theory to determine a rate for the transition based on the difference in internal energy between states. The rate

of migration from state i to state j passing through a saddle point s is given by

$$r_{i\to j} = \left(\frac{3}{5}\right)^{1/2} \nu_D \exp\left(-\frac{\langle c\rangle\, \Omega \Delta^2}{k_B T}\right) \exp\left(-\frac{E_j - E_i}{2k_B T}\right) \tag{24}$$

where E_i is the internal energy of the system in state i, ν_D is the Debye frequency and Ω the volume of the migrating atom [17]. We have taken the Debye frequency to be a constant value $\nu_D = 1 \times 10^{13} s^{-1}$ for all simulations. The parameter Δ is the displacement, expressed as a fraction of the bond length, at which the force on the migrating atom is a maximum, and is taken to be the constant value $\frac{1}{\sqrt{10}}$. $\frac{1}{\langle c\rangle}$ is an effective elastic compliance. For fcc crystals this is given by [17]:

$$\frac{1}{\langle c\rangle} \approx \frac{2}{15}\left(\frac{3}{c_{11}} + \frac{2}{c_{11} - c_{12}} + \frac{1}{c_{44}}\right) \tag{25}$$

Only $\frac{1}{2} < 110 >$ hops are considered. Saddle points for exchanges with the next-nearest neighbour shell of atoms are around 2eV higher in energy and so are assumed not to occur. We have further assumed that correlated movements of atoms, such as may be seen in surface diffusion[18] are not present here.

8.2 Number of flickers per move

In figure 1, we plot the number of flickers per move as a function of the real time simulated. Each line represents the time- average of twelve independent runs. This averaging was performed by collecting the mean number of flickers per move over a period of 1000 chains of events, as defined in section 6.1, together with the real time represented by these moves. These means were then gathered into bins spaced on the logarithmic real time axis. The lines shown in figure 1 are to guide the eye. Each run took roughly the same computational time, and represents the same number of new states visited (1 million). However, as the time taken per move is environmentally dependent, the real time simulated is not the same for all runs, and is orders of magnitude different for the alloys considered. Note that the graph is plotted on a log-log scale. As all quantities are determined stochastically, if one run generates a very large number of flickers or long time steps, it will tend to dominate the average.

In pure aluminium, we expect for two widely separated vacancies $p_{\mathrm{j}\to\mathrm{k}} = \frac{1}{24}$,$\gamma_{\mathrm{j}\leftrightarrow\mathrm{k}} = \frac{1}{24^2}$ and so $\gamma_{\mathrm{j}(0)} = \frac{1}{24}$. The expected number of flickers is given by equation (6). In this case for a pure second order calulation we expect 0.0417 flickers per move if a reversal event is defined and 0.0435 flickers per move otherwise. This latter value is shown by the bold line in figure 1. These numbers change only when the two vacancies come together. In alloys the probability of each event occurring must be calculated for each particular atomic configuration.

We see in figure 1 that the pure aluminium simulation has considerably fewer flickers per

move than the horizontal line corresponding to the second order RTMC algorithm. This result is due to the fact that we are considering flickers only to previously visited states, and as the vacancies move around they are exploring new regions of configurational phase space. The alloys were all found to have more flickers per move than the value predicted by the second order RTMC algorithm, indicating that unproductive reversal moves are indeed problematic in alloys.

It can be seen that all alloys considered start at roughly the same number of flickers per move, about 0.5. The number of flickers per move for alloy compositions with more magnesium added (alloys 4 and 5) are seen to increase from this value, to 100 and even 1000 flickers per move on average. When such a large number of flickers precedes each move, the real time clock can be advanced substantially for each move, and the total real time simulated is greater. Note that both vacancies must be flickering here. If one vacancy is free to wander around the crystal, this will tend to be a higher rate process than those contributing flickering moves, and so only when all vacancies are trapped are large numbers of flickers seen.

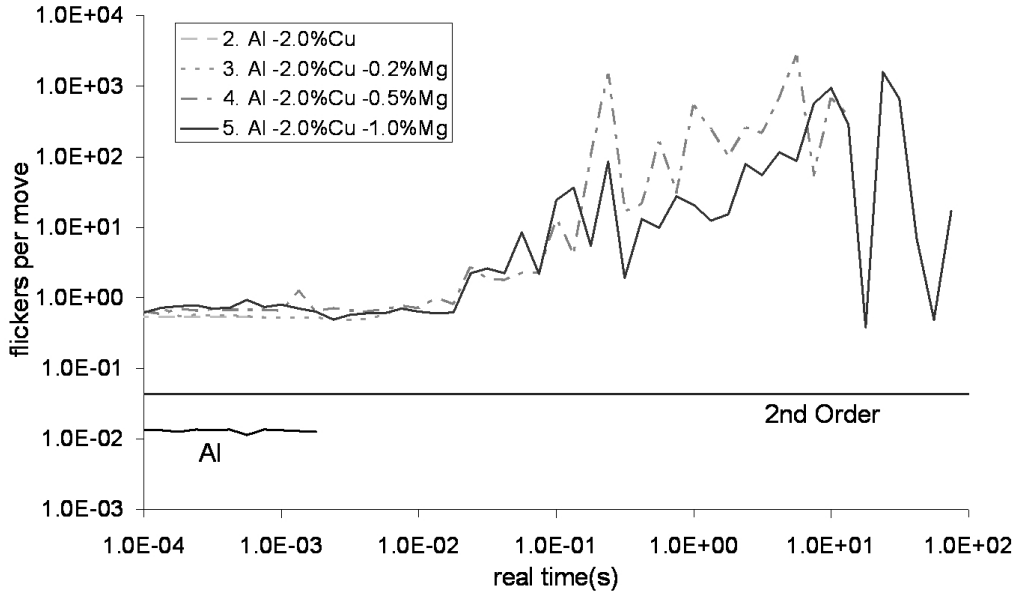

Fig. 1. The number of flickers per move as a function of the real time simulated, calculated using the chain RTMC algorithm described in section 6.1. It can be seen that the real time simulated for alloy compositions 4 and 5 (having 0.5% and 1.0% Mg respectively) is orders of magnitude greater than that for the others. Also in these simulations we see an increase from under one flicker per move to over 100 at later times. The solid horizontal line marks the expected number of flickers per move for the second order RTMC algorithm with two vacancies in a homogeneous medium.

8.3 Number of moves per chain

In figure 2 we plot the number of moves per chain, that is to say the number of moves which occur from a given starting state before an event is selected to take the system into a state about which there is only first order information available.

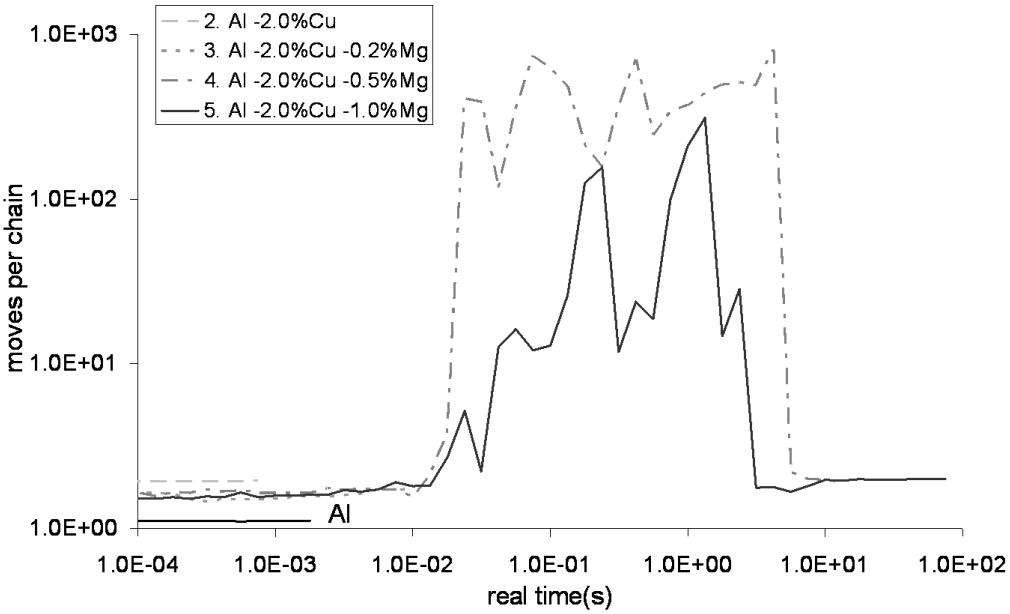

Fig. 2. The number of moves per chain as a function of the real time simulated. Note that at greater times, the vacancies in alloy compositions 4 and 5 (0.5% and 1.0% Mg) are trapped and so perform hundreds of moves before the system moves on from a local region of configurational phase space.

There is a minimum possible value of one event per chain, and it is seen that the pure aluminium simulation remains close to this value, suggesting that the system is exploring phase space rather than retracing its steps. Alloys 2 and 3 (low magnesium) perform roughly two events per chain on average. The high magnesium alloys (4 and 5) are seen at later times to be performing hundreds of events on average for each chain constructed.

8.4 Number of events per rate calculation

In figure 3 we plot the average number of events which occur per rate calculation required. A rate calculation is deemed necessary whenever an event takes the system across a saddle point not previously visited. Note that in the special case when the saddle point is a function only of the two states it joins, this will be an overestimate of the number actually performed. The number of events per rate calculation is an estimate of the efficiency of the chain RTMC algorithm. If a first order simulation without a history of previously visited states were to

be performed, we should expect to have to perform 23 rate calculations before determining the next event. With the chain RTMC algorithm many flickers and long chains of events can be performed when the system is trapped in a region of phase space.

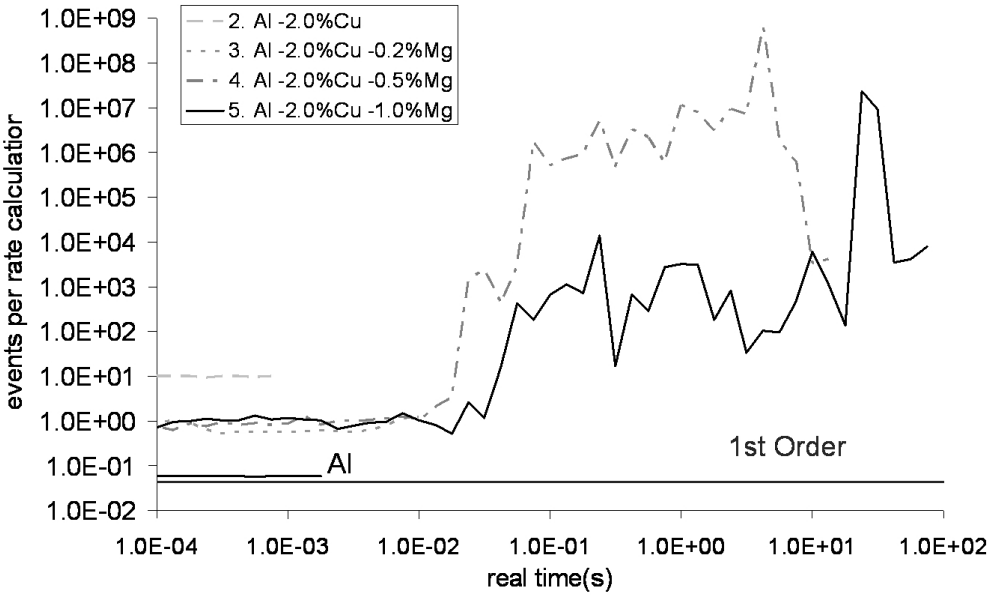

Fig. 3. The number of events per rate calculation as a function of the real time simulated. This is a measure of the efficiency of the algorithm. The horizontal line represents the expected number of events performed per rate calculation if a first order residence time algorithm is used ($\frac{1}{23}$). Note that all the alloy systems considered perform better, and that when magnesium is added (alloys 3,4,5) the chain RTMC algorithm is orders of magnitude more efficient.

The pure aluminium system once again shows least advantage of the chain RTMC algorithm over the first order residence time algorithm, as it is only when a system retraces its steps through phase space that efficiency improvements can be made. All the alloy systems are an order of magnitude more efficient in terms of the number of required rate calculations. In particular we note that the Al -2at.%Cu system runs steadily at ten events per rate calculation, and that alloys 4 and 5 reach 1000 and 1×10^5 events per rate calculation respectively.

9 Summary and Conclusions

The problem of flickering is endemic to kinetic Monte Carlo modelling, and it is particularly onerous when the calculation of the transition rates is expensive and must be done on the fly. We have presented several techniques designed to permit simulation of systems prone to flicker.

In section 2 we described the n-fold way as a stochastic first order residence time algorithm highlighting the difficulties that may arise with getting trapped at metastable states, which we called the flicker problem. The second order algorithm of [1] was introduced to overcome the flicker problem and it was rederived in section 3 to enable a stochastic implementation of the second order residence time algorithm as described in section 4. Practical difficulties with the second order residence time algorithm were identified in section 5.

This led us to develop in section 6 a new algorithm in which the stochastic second order residence time algorithm is dynamically invoked. We showed that this combined first and second order algorithm is never less efficient than the n-fold way algorithm. Although this is readily generalized to higher order algorithms it would be at the expense of increased storage requirements. Instead we found it is more efficient to store the recent history of visited states. If one has used the second order algorithm in a situation where events are independent of each other then one can reuse rate information about these events in the chain algorithm of section 6. This enables the system to be evolved without having to calculate any more rates.

The efficiency gain possible with the chain RTMC algorithm is shown in figure 3, where we see that the number of rate calculations required to perform each Monte Carlo move is fewer than that required by a simple first order algorithm. Indeed we show that for the Al -2at.%Cu system simulated we can perform 10 events per rate calculation using the chain algorithm, as opposed to $\frac{1}{23}$ with the first order algorithm. The effectiveness of the chain RTMC algorithm becomes particularly significant where large numbers of flickers are expected. We have shown that when magnesium is added to the Al- Cu system, this has the effect of trapping vacancies. Our algorithm can perform thousands of events between rate calculations in these systems, achieving real simulated times significantly longer than those achievable with either the first or second order residence time algorithms.

10 Glossary

direct and indirect moves If the system moves from state j to state k and then does not immmediately return to state j the move is called direct. If the system makes one or more flickers before escaping from state j the move is called indirect.

event An event moves the system between states. Event k takes the system from whatever state it currently occupies into state k. An event is indicated by an italic character e.g. i, j, k.

first and second order information The rate $r_{\mathrm{j}\to\mathrm{k}}$ at which event k occurs from state j is the first order information. The rate $r_{\mathrm{k}\to\mathrm{l}}$ at which a subsequent event event l occurs is second order information when viewed from state j.

flicker A flicker between state i and state j is composed of two events. First event j takes the system from state i to state j. Then event i takes the system back to state i.

influence set The influence set $\mathcal{K}$ of event k contains all the events corresponding to physical processes affected by moving to state k.

rate The rate of moving the system from state i to state j is represented as $r_{\mathrm{i}\to\mathrm{j}}$. The rate is the inverse of the average time taken for an ensemble of systems in state i to move to

state j.

reliability level The reliability level is an indicator of the level of accuracy associated with a calculation. Estimates of reliability level 1 may be directly compared with each other, as may estimates of 2a or 2b, but a level 1 may not be compared to a level 2.

residence time Each event is an independent Poisson arrival. The residence time is the length of time waited before an event occurs.

reversal event The reversal event takes the system to the previously visited state.

state A state is a point in the configurational phase space of the system. A state is indicated by a roman character eg i, j, k.

site A site is a location of an atom. A site is indicated by an uppercase character eg I, J, K.

11 Acknowledgements

D.R.M was supported by an EPSRC grant. The computer simulations were performed on Oswell, the Oxford University Supercomputer. This work was performed in part under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livermore National Laboratory, under Contract No. W-7405-Eng-48. D.R.M. gratefully acknowledges the fellowship in the LLNL Summer Institute on Computational Materials Science and Chemistry through which part of this work was done.

References

[1] M. Athènes, P. Bellon, and G. Martin. Identification of novel diffusion cycles in B ordered phases by Monte Carlo simulation. *Phil. Mag. A*, 76:565, 1997.

[2] A.B. Bortz, M.H. Kalos, and J.L. Lebowitz. A new algorithm for Monte Carlo simulation of Ising spin systems. *J. Comp. Phys.*, 17:10–18, 1975.

[3] K.A. Fichthorn and W.H. Weinberg. Theoretical foundations of dynamical Monte Carlo simulations. *J. Chem. Phys.*, 95(2):1090–1096, 1991.

[4] T.T. Rautiainen and A.P. Sutton. Influence of the atomic diffusion mechanism on morphologies, kinetics, and the mechanisms of coarsening during phase separation. *Phys. Rev. B*, 59(21):13681–13692, 1999.

[5] T.T. Rautiainen. *Modelling microstructural evolution in binary alloys.* D.Phil. thesis, University of Oxford, 1999.

[6] R.Weinkamer and P.Fratzl. By which mechanism does coarsening in phase-separating alloys proceed? *Europhys. Lett.*, 61(2):261–267, Jan 2003.

[7] C.S. Deo and D.J. Srolovitz. First passage time Markov chain analysis of rare events for kinetic Monte Carlo: double kink nucleation during dislocation glide. *Modelling Simul. Mater. Sci. Eng.*, 10:581–596, 2002.

[8] W. Cai, M. H. Kalos, M. de Koning, and V. V. Bulatov. Importance sampling of rare transition events in Markov processes. *Phys. Rev. E*, 66:046703, 2002.

[9] M.A. Novotny. Monte Carlo with absorbing Markov chains: fast local algorithms for slow dynamics. *Phys. Rev. Lett.*, 74:1–5, 1995.

[10] G. Korniss, Z. Toroczkai, M.A. Novotny, and P.A. Rikvold. From massively parallel algorithms and fluctuating time horizons to nonequilibrium surface growth. *Phys. Rev. Lett.*, 84(6):1351–1354, 2000.

[11] G. Korniss, M.A. Novotny, and P.A. Rikvold. Parallelization of a dynamic monte- carlo algorithm: A partially rejection- free conservative approach. *J. Comp. Phys.*, 152:488–503, 1999.

[12] B. Lubachevsky and A. Weiss. Synchronous relaxation for parallel Ising spin simulations. In *Proceedings of the 15th Workshop on Parallel and Distributed Simulation*. Institute of Electrical and Electronics Engineers Inc., 2001.

[13] B. Lubachevsky. Efficient parallel simulations of asynchronous cellular arrays. *Complex systems*, 1:1099, 1987.

[14] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in Fortran*. Cambridge University Press, 2nd edition, 1992.

[15] M.W. Finnis and J.E. Sinclair. A simple empirical n- body potential for transition metals. *Phil. Mag. A*, 50:45–55, 1984.

[16] G.H. Vineyard. Frequency factors and isotope effects in solid state rate processes. *J. Phys. Chem. Solids*, 3:121–127, 1957.

[17] C.P. Flynn. *Point Defects and Diffusion*. Clarendon Press, 1972.

[18] J.A. Sprague, F. Montalenti, B.P. Uberuaga, J.D. Kress, and A.F. Voter. Simulation of growth of Cu on Ag(001) at experimental deposition rates. *Phys. Rev. B*, 66:205415, 2002.

[19] A.P. Sutton and J. Chen. Long- range Finnis- Sinclair potentials. *Phil. Mag. Lett.*, 61:139–146, 1990.

[20] F. Stillinger and T.A. Weber. Computer simulatoin of local order in condensed phases of silicon. *Phys. Rev. B*, 31:5262–5271, 1985.

[21] F. Stillinger and T.A. Weber. *Phys. Rev. B*, 33:1451, 1986.

[22] H. Rafii-Tabar and A.P. Sutton. Long- range Finnis- Sinclair potentials for fcc metallic alloys. *Phil. Mag. Lett.*, 63:217–224, 1991.

[23] J.I. Takumara. *Physical Metallurgy*. North-Holland, Amsterdam, 1965.

[24] A. Cottrell. *An Introduction to Metallurgy*. Institute of Materials, 2nd edition, 1995.

[25] D.R. Lide (Ed.). *CRC Handbook of Chemistry and Physics*. CRC Press, New York, 76 edition, 1995.

A Interatomic potentials

An analytic form for the interatomic potentials based on that used by Sutton and Chen [19] is used with an additional exponential term from the Stillinger-Weber potential[20,21] to ensure the potentials and all their derivatives smoothly tend to zero at the cut-off radius. The potential energy of the system is given by:

$$E = \tfrac{1}{2}\sum_i \sum_j v\left(r_{ij}\right) - \sum_i \left[\sum_j \phi\left(r_{ij}\right)\right]^{\frac{1}{2}} \tag{A.1}$$

$$v\left(r_{ij}\right) = a_{ij}\left(\frac{r_{ij}}{r_0}\right)^{-m_{ij}} \exp\left[\frac{c_{ij}}{r_{ij}-x}\right] \tag{A.2}$$

$$\phi\left(r_{ij}\right) = b_{ij}\left(\frac{r_{ij}}{r_0}\right)^{-n_{ij}} \exp\left[\frac{c_{ij}}{r_{ij}-x}\right] \tag{A.3}$$

r_{ij} is the distance between atoms i and j. $a_{ij}, b_{ij}, c_{ij}, m_{ij}$ and n_{ij} are positive parameters and r_0 is the unit of length, which we take to be 1 Å . x is the cutoff radius, taken here to be 4.5Å for all atom pairs considered, which is roughly midway between second and third neighbours in the aluminium matrix. The particular parameterisation of the elemental potentials used is given in table A.1. The parameters quoted for Mg give an hcp structure as the ground state, with an ideal c/a ratio and a=3.20 Å . However, this structure has the same energy as an fcc structure with a lattice parameter of a=4.53 Å . The degeneracy of these crystal structures is a consequence of the range of atomic interactions extending only to first neighbours in both structures in Mg. The five elastic constants of hcp Mg were fitted in a least squares sense, and the values quoted in Tables A.1 and A.2 are those calculated for fcc Mg.

If r_{ij} links atoms of different types, the following prescription [22] is used to generate intermediate forms for the parameters:

$$\begin{aligned}
a_{ij} &= \left(a_{ii} \times a_{jj}\right)^{\frac{1}{2}} \\
b_{ij} &= \left(b_{ii} \times b_{jj}\right)^{\frac{1}{2}} \\
c_{ij} &= \tfrac{1}{4}\left(\tfrac{c_{ii}}{l_{ii}} + \tfrac{c_{jj}}{l_{jj}}\right)\left(l_{ii} + l_{jj}\right) \\
m_{ij} &= \tfrac{1}{2}\left(m_{ii} + m_{jj}\right) \\
n_{ij} &= \tfrac{1}{2}\left(n_{ii} + n_{jj}\right)
\end{aligned} \tag{A.4}$$

These potentials are unlikely to reproduce energy differences between intermetallic structures, and hence they are unlikely to reproduce the phase diagram, which is very complex. Nevertheless, they suit our need by providing a means of estimating relaxation forces and energies for arbitrary alloy configurations, while reproducing the elastic properties of the pure metals. In addition, Monte Carlo simulated annealing runs of the $\mathrm{Al-Cu}$ system at zero

	units	Al	Cu	Mg
a	eV	340.54	1167.46	429.89
b	eV^2	178.67	138.82	1.774
c	Å	0.321	1.591	2.313
m		6.566	8.305	6.027
n		4.084	3.748	0.000

Table A.1
Parameterisation of the potentials for the pure metals Al, Cu and Mg.

	units	Al	Cu	Mg
E_{coh}	eV	3.34 (3.34)	3.50 (3.50)	1.50 (1.50)
c_{11}	GPa	98.3 (108)	170 (170)	52.1 (c_{11}=59.3,c_{33}=61.5)
c_{12}	GPa	61.6 (62.2)	123 (123)	29.5 (c_{12}=25.7,c_{12}=21.4)
c_{44}	GPa	29.8 (28.4)	75.3 (75.3)	22.6 (16.4)
l	Å	4.05 (4.05)	3.61 (3.61)	4.53 (a=3.20,c=5.20)
E_{vac}	eV	0.824 (0.78)	1.129 (1.07)	0.537
E_{mig}	eV	0.656 (0.56)	0.749 (0.88)	0.568

Table A.2
The potentials were fitted to five physical constants: the cohesive energy, three elastic constants and the lattice parameter of the fcc unit cell. The fitted values are shown in the table above. Also shown in this table are the vacancy formation energy (calculated for the atomistically relaxed structure) and migration barrier (calculated using Flynn's prescription in equation (24) for the pure metal. Experimental values are shown in parentheses. The experimental values for the vacancy formation and migration energies are from [23]. The elastic constants are from Cottrell[24], the lattice parameters and cohesive energies from [19,25]. Note that experimental values for the vacancy formation energy and migration barrier do not exist for the hypothetical fcc Magnesium.

external stress showed that our potentials favour intermetallic phases at the compositions Al_3Cu , AlCu and Cu_3Al with crystal structures $L1_2$, $L1_0$ and $L1_2$ respectively.